ESD-TR-83-024

Semiannual Technical Summary

Restructurable VLSI Program

31 March 1983

Prepared for the Defense Advanced Research Projects Agency
under Electronic Systems Division Contract F19628-80-C-0002 by

Lincoln Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Lexington, Massachusetts

Approved for public release; distribution unlimited.

The work reported in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. This work was sponsored by the Defense Advanced Research Projects Agency under Air Force Contract F19628-80-C-0002 (ARPA Order 3797).

The views and conclusions contained in this document are those of the contractor and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the United States Government.

The Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

Thomas J. Alpert

Thomas J. Alpert, Major, USAF
Chief, ESD Lincoln Laboratory Project Office

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY

RESTRUCTURABLE VLSI PROGRAM

SEMIANNUAL TECHNICAL SUMMARY REPORT
TO THE
DEFENSE ADVANCED RESEARCH PROJECTS AGENCY

1 OCTOBER 1982 — 31 MARCH 1983

ISSUED 10 JUNE 1983

Approved for public release; distribution unlimited.

LEXINGTON MASSACHUSETTS

ABSTRACT

This report describes work performed on the Restructurable VLSI Program sponsored by the Information Processing Techniques Office of the Defense Advanced Research Projects Agency during the semiannual period 1 October 1982 through 31 March 1983.

TABLE OF CONTENTS

	Abstract	iii
	List of Illustrations	vi
	List of Tables	vii
I.	PROGRAM OVERVIEW AND SUMMARY	1
II.	DESIGN AIDS FOR RVLSI	7
	A. MACPITTS	7
	B. LBS	8
	C. Hierarchical Chip Assembler	10
	D. Laser Zapping Facility	13
	E. Placement, Assignment, Linking	15
III.	APPLICATIONS	17
	A. Dynamic Time Warping (DTW) Wafer	17
	B. AGC Circuit	21
IV.	TESTING	35
References		45
APPENDIX A - BNF for LBS Topological Intermediate Form		47
APPENDIX B - MACPITTS Program for AGC Circuit		49

LIST OF ILLUSTRATIONS

Figure No.		Page
1	DTW Wafer with 63 Processing Elements	18
2	DTW Processing Element	20
3	LPC Vocoder with AGC	22
4	AGC IC Floor Plan	26
5	Photomicrograph of AGC Chip	27
6	AGC IC Pin-Out and Peripheral Circuitry for Use in the LPC Vocoder	29
7	AGC IC Input/Output Sample-Rate ("sample-stuff") and Frame-Rate ("frame-stuff") Processing Timing	30
8	Fragment of AGC MACPITTS Source Code	31
9	Flowchart for Sample-Rate Task of AGC Algorithm	32
10	TOC System Block Diagram	36
11	TOC Memory Banks	38
12	TOC Vector Encoding	39
13	TOC Data Flow	40
14	TOC Four-Pin Functional Tester Slice	43

LIST OF TABLES

Table No.		Page
I	Master-Slave Flip-Flop with Two-Phase Clocks Generated Externally	9
II	Master-Slave Flip-Flop with Two-Phase Clocks Generated Internally	11

RESTRUCTURABLE VLSI PROGRAM

I. PROGRAM OVERVIEW AND SUMMARY

A. OVERVIEW

The main objective of the Lincoln Restructurable VLSI Program (RVLSI) is to develop methodologies and architectures for implementing wafer-scale systems with complexities approaching a million gates. In our approach, we envisage a modular style of architecture comprising an array of cells embedded in a regular interconnection matrix. Ideally, the cells should consist of only a few basic types. The interconnection matrix is a fixed pattern of metal lines augmented by a complement of programmable switches or links. Conceptually, the links could be either volatile or nonvolatile. They could be of an electronic nature, such as a transistor switch, or could be permanently programmed through some mechanism such as a laser. The RVLSI Program is currently focusing on laser-formed interconnect.

The link concept offers the potential for a highly flexible, restructurable type of interconnect technology that could be exploited in a variety of ways. For example, logical cells or subsystems found to be faulty at wafer-probe time could be permanently excised from the rest of the wafer. The flexible interconnect could also be used to "jump around" faulty logic and tie in redundant cells judiciously scattered around the wafer for this purpose. Also, the interconnect could be tailored to a specific application in order to minimize electrical degradations and performance penalties caused by unused wiring.

Further, the testing of a particular logical subsystem buried deep within a complex wafer-scale system poses a very difficult problem. A properly designed restructurable interconnect matrix could be temporarily configured to render internal cells both controllable and observable from the wafer periphery. In this way, each component cell or a tractable cluster of cells could be tested in straightforward manner using standard techniques. With an electronic linking mechanism it is possible to think in terms of a dynamically reconfigurable system. Such a feature could be used to alter the functional mode of a system subject to changes in the operating scenario, or it could be used to support some degree of fault tolerance if the system architecture was suitably designed.

Several major areas of research have been identified in the context of the RVLSI concept:

(1) System architectures and partitionings for whole-wafer implementations.

(2) Placement and routing strategies for optimal utilization of redundant resources and efficient interconnect.

(3) Assignment and linking algorithms to exploit redundancy and flexible interconnect.

(4) Methods for expediting cell design with emphasis on functional level descriptions, enhanced testability, and fault tolerance.

(5) Methods for testing complex, multiple-cell, whole-wafer systems.

Complementary work on the development of various link and interconnect technologies as well as fabrication/processing technology is being supported by the Lincoln Air Force Line Program, and results are reported under the Lincoln Laboratory Advanced Electronic Technology Quarterly Technical Summary.

B. SUMMARY OF PROGRESS

Work for this period is reported under three headings: Design Aids for RVLSI (Sec. II), Applications (Sec. III), and Testing (Sec. IV). The following paragraphs summarize progress to date.

1. Design Aids for RVLSI

The current NMOS version of the MACPITTS system has been readied for external distribution. A user's manual has been written and a substantial amount of work has been completed on internal maintenance documents. As of this date, the source for MACPITTS has been transmitted to three external sites:

One design project, the TOC (tester-on-chip) circuit, has stressed the layout capabilities of MACPITTS. In order for chips of this complexity to be fabricated by MOSIS, the layout must be more compact. Several organelles were redesigned to reduce their size. Also, a 3-μm pad library was added, allowing TOC to be fabricated at 3 μm. Changes were also made to the pad distribution algorithm to afford a better allocation of pads among the chip sides.

Several new designs have been fabricated. Two were simple organelle test chips that will be used to check the performance of specific organelles. They are in the process of being tested. The third project fabricated was a vocoder AGC (Automatic Gain Control) chip. Of nine AGC chips fabricated, all passed functional tests. The tests were run at 400 kHz, which is double the required system clocking rate. More detailed dynamic performance tests are under way. Other circuits currently in design are a 3-4-5 FIR digital filter, an 8-bit multiplier, and a CORDIC (Coordinate Rotation Digital Computation) rotator.

The LBS (Lincoln Boolean Synthesizer) designed chips from the first MOSIS 5-μm CMOS run have arrived. The designs were immediately tested for DC functionality, which they substantially passed. The most complex of these was a 4-bit ALU modeled after the TTL 74181. A minor error in the LBS compiler was, however, discovered. Due to the unavailability of the Tektronix S3260 tester, more complete testing was temporarily postponed but is now in progress. Preliminary tests demonstrate the operation of some of the master-slave flip-flop designs at better than 10 MHz. The ALU operated at 2 MHz.

Work on the second version of LBS is proceeding. A preliminary LBS topological intermediate format has been agreed on, and a user-interface which implements a macro expansion and some logic compaction has been implemented. Further refinement is needed, however.

A system for interconnecting cell substructures in order to assemble larger macro structures is well under way. Definition of the preliminary system capabilities was completed considering time constraints on implementation and the future incorporation of more advanced features. The initial system supports manual placement of polygon-shaped cells and partial interconnect, which will allow the user control over critical parts of the design and automatic routing of signals.

Two areas of work have been proceeding simultaneously:

(a) A CAESAR-based graphics interface to specify placement of cells, pin locations, and partial interconnect.

(b) High-level global routing scheme to reduce the general problem to a set of channel routing problems.

In the area of applications, a "mixed master slice" gate array chip d´ ign is being studied.

The Restructurable Wafer Editor (RWED) can now command the IEEE-488 Instrumentation Bus attached to the laser controller. This is the first step toward an interactive test/zap capability at the laser table. Schemes have been developed which use this capability for evaluating the interconnect of a fresh RVLSI wafer and for incrementally restructuring the interconnect while checking for certain kinds of faults.

RWED has been successfully ported to the second laser table, which is now fully operational. This system is currently being used for optical probing experiments.

A set of RVLSI wafer design programs developed under a cooperating program, including a generalized linker, has been adapted for the packet radio integrator application. The linker is relatively insensitive to specifics of the interconnect topology and the entire system is particularly well suited to structures which comprise mainly one-cell type with a small number of special peripheral cells at the wafer edges. The system includes means for describing the logical wafer structure, the general physical wafer structure, a particular physical wafer structure (i.e., unique defect pattern), and actions necessary to restructure a wafer.

2. Applications

An automatic gain control circuit for a narrowband vocoder has been designed, fabricated, and tested using MACPITTS. The chip contains 2190 transistors, occupies 4.5×5.4 mm^2, and was fabricated in 4-μm NMOS. A variety of tests have been performed or are in progress to quantify the chip's performance limits. One AGC chip has been evaluated in the LPC vocoder test bed for which it was designed. The use of the AGC chip in the Lincoln compact LPC vocoder (developed under the DARPA Packet Speech Systems Technology Program) marks the first use of a MACPITTS generated chip in an actual field application.

A systolic array architecture has been developed to solve the computationally intensive search/match portion of the isolated and connected word-recognition problem: dynamic time warping (DTW). The array, intended for restructurable wafer-scale implementation in Lincoln's 5-μm bulk CMOS process, is projected to consist of 63 custom-designed processing elements utilizing a total of 350,000 functioning transistors. A bit-serial arithmetic approach is being pursued for its high throughput property and because it results in physically smaller cells and less interconnect than would be possible with parallel arithmetic designs. The wafer will be part of an advanced speech recognition front end and can be used for a >100-word speaker independent isolated or connected word-recognition applications. It represents an order of magnitude increase in computational power over similarly sized, commercially available systems. A flexible systolic system simulator has been developed in LISP and is currently being used for detailed logical design, and simulation and verification of the array architecture. The effort will soon be transitioned to layout.

3. Testing

The TOC-tester design has been revised and only one unique custom chip design is now needed. This chip has been designed using MACPITTS and thoroughly simulated. In order to meet MOSIS maximum die size constraints, several new, ultra compact organelles were added to MACPITTS. Using a new Lincoln-developed CAESAR-to-L5 interface (graphics-to-text), the layout effort required only two weeks. This resulted in the reduction of 42 data path units to 29. The circuit was submitted to the 21 March 1983 MOSIS 3-μm NMOS run. It comprises 4810 transistors and occupies 6.4 × 7.2 mm^2. It required 216 lines of code to generate, is expected to dissipate 1 W, and should operate at about 2 MHz.

II. DESIGN AIDS FOR RVLSI

A. MACPITTS

During the first half of FY 83, the initial NMOS version of the MACPITTS system[1] was readied for external distribution. A user's manual was written and work was begun on internal maintenance documents. As of this date, the source for MACPITTS has been sent to three external sites. During the past 6 months, a number of improvements and fixes were made in response to in-house users. Errors uncovered and corrected include a faulty register driver circuit and problems in the resource extraction process and routing of power and ground. Other improvements were made to allow fabrication of very large design projects. One design project, the TOC (tester-on-chip, see Sec. IV), has especially stressed the layout capabilities of MACPITTS. In order for TOC to be fabricated, the layout had to be reduced in size. Several organelles were redesigned to make them more compact. Also, a 3-μm pad library was added, allowing TOC to be processed at 3 μm. Changes were also made to the pad distribution algorithm to afford a better allocation of pads among the chip sides.

It was recognized for a long time that it can be a slow process to lay out small pieces of circuitry using textual layout language like L5 (Ref. 2). The very quality which allows L5 to be so powerful can also be a detriment. It is much easier to use a graphical editor like CAESAR[3] to build these small pieces. Therefore, an interface was developed between L5 and CAESAR. This allows graphical design of nonparameterized layouts which can then be converted into L5 subroutines and called from higher level programs. This system has been used to design several new organelles and redesign the register drivers.

Several designs were fabricated during the first semester. Two were simple test chips that will be used to check the performance of specific organelles. These are in the process of being tested. The third project fabricated was a vocoder AGC chip (Automatic Gain Control, see Sec. III-B).

Of nine chips fabricated, all passed functional tests. The tests were run at 400 kHz which is double the required system clocking rate. More detailed performance tests are beginning.

In-house use of MACPITTS is increasing as confidence grows based on previous successes. Among the circuits being designed are a 3-4-5 digital FIR filter, an 8-bit multiplier, and a CORDIC (short for Coordinate Rotation Digital Computation) rotator. As with the AGC, the designers of these circuits have learned how to use MACPITTS, coded and simulated their respective algorithms, and generated CIF in a short period of time.

B. LBS

The LBS-designed chips from the first MOSIS CMOS run arrived during this quarter and comprised 7 designs. Five of the designs were immediately tested for DC functionality, which they substantially passed. A minor error in the LBS compiler was, however, discovered. More complete testing with the Tektronix 3260 tester is now in progress. Test results are as follows.

1. Master-Slave Flip-Flops with Two-Phase Clocks Generated Externally

These master-slave flip-flops use two-phase clocks generated externally. Each chip has four flip-flops with fully independent inputs and outputs. In the test jig, flip-flops 2, 3, and 4 are connected as a shift register by external jumpers; flip-flop 1 is independent.

There are two sets of chips, five in each set. These sets were labeled M27QN4 and M20YN4 by MOSIS. We labeled the chips a-e and f-j, respectively. Chip h did not work. The M27QN4 chips operated in the shift register configuration at an average of 9 MHz. The M20YN4 chips operated at an average of 7 MHz. The independent flip-flop operates a consistent 20 ns faster (11 and 8 MHz, respectively). The difference is probably due to I/O pad delay, so an internally connected shift register should operate at the independent flip-flop rate (see Table I).

TABLE I

MASTER-SLAVE FLIP-FLOP WITH TWO-PHASE CLOCKS GENERATED EXTERNALLY

Chip	Functional	Shift Register Clock Period (ns)	Single Flop Clock Period (ns)
a	yes	160	-
b	yes	80	60
c	yes	120	100
d	yes	120	-
e	yes	80	-
f	yes	120	-
g	yes	160	-
h	no	-	-
i	yes	140	-
j	yes	140	-

2. Master-Slave Flip-Flop with Observable Two-Phase Clocks Generated Internally from a Single External Clock

These master-slave flip-flops use observable two-phase clocks generated internally from a single external clock. Each chip has four flip-flops with fully independent inputs and outputs. In the test jig, flip-flops 2, 3, and 4 are connected as a shift register by external jumpers; flip-flop 1 is independent.

The internally generated two-phase clocks are brought out via output pins and therefore are observable. The clock generation circuitry is modeled after the circuit on page 233 of Ref. 4.

There are two sets of chips, five in each set. These sets were labeled M27QN3 and M20YN3 by MOSIS. We labeled the chips a-e and f-j, respectively. Neither set is particularly fast. The M27QN3 chips, however, operate at about 1.67 MHz, while the M20YN3 chips operate at 576 kHz. There are also substantially more failures with the M20YN3s. However, their power drain is 10 percent of M27QN3s (see Table II).

Work on the second version of LBS is proceeding. A preliminary LBS topological intermediate format has been agreed upon, and a top-level user interface has been implemented. The present topological intermediate form between the top-level or user-interface part and the layout part is defined in Bakus-Naur Form in Appendix A.

C. HIERARCHICAL CHIP ASSEMBLER

At this moment, a large collection of tools to design integrated circuits is available, ranging from systems to do hand layout to highly sophisticated circuit synthesizers. Little effort has been devoted to the very real problem of producing a circuit by putting together pieces developed in a variety of ways using these different design aids.

The Chip Assembler will address this problem, allowing the design of very large scale ICs by integrating the different parts in a hierarchical

TABLE II

MASTER-SLAVE FLIP-FLOP WITH TWO-PHASE CLOCKS
GENERATED INTERNALLY
(Two-Phase Clocks Output for Observation)

Chip	Functional		Shift Register Clock Period (ns)	Single Flop Clock Period (ns)	Current at Shift Register Speed (mA)
	Shift Register	Single Flop			
a	no	no	-	-	-
b	yes	yes	600	500	4.51
c	yes	yes	600	500	5.75
d	yes	yes	600	500	9.94
e	yes	-	600	-	4.89
f	yes	yes	2500	2500	0.31
g	no	yes	1500	-	
h	yes	yes	2000	-	
i	yes	yes	2000	1000	0.61
j	yes	yes	2500	2000	0.50

fashion. As such, it is not intended to be used to design at the transistor level, but to deal with cells of increasing levels of complexity in a way that only a minimum amount of information needs to be carried over.

The Chip Assembler is a multilevel placement and routing system. Its main features are:

(1) A simple description of cells based on polygonal outlines and input/outputs entered using a CAESAR-based graphics editor. Only at the lowest levels is reference to complete description (CIF files) required. Changes in cells do not necessarily impact other levels, thus simplifying the design process. Initially, the user manually specifies the placement of cells.

(2) Interconnection of cells is described through net lists with the option to add physical data to form part of a net. This allows direct intervention of the designer in the routing of the signals, thereby controlling critical paths and improving the performance of automatic routers.

(3) Two-phase automatic routing of signals. First a global "loose" routing is performed, testing the feasibility of completing the interconnections given certain free areas. Then the problem is reduced to a set of independent routing tasks inside rectangles, a much better understood case.

The development of the system draws on the extensive research done in the areas of placement and routing for VLSI chips. The modularity of the current approach will facilitate refinement and optimization of the different components.

D. LASER ZAPPING FACILITY

The Restructurable Wafer Editor program (RWED) can now command the IEEE-488 Instrumentation Bus attached to the laser controller. This is the first step toward interactive restructurability at the laser table.
Two significant changes were made to the system to accomplish this:

(1) The VAX-controller protocol was expanded so that the controller could initiate messages to the screen (and record file).

(2) Another command was added to this protocol from which the controller parses the appropriate IEEE-488 subcommands and arguments.

The RWED command for controlling the bus is: l<c><arg>...

Where <c> is the single letter IEEE-488 Bus function, and the <arg>s are its arguments. There are two classes of IEEE-488 Bus subcommands, passive and active. Passive commands simply cause a function to happen. Active commands imply a response. This response takes the form of a controller-generated message, both displayed on the operator's window, and written in the record file. These subcommands are:

K	Sends the local lockout universal command.
U	Sends the universal device clear.
C	Sends selected device clear to the specified device.
L	Sets the specified device to local mode.
W	Causes a pause of <delay> milliseconds. This is not really an IEEE-488 bus command but comes with the driver.
T	Sends selected device trigger and accepts response. The response will appear as a controller-generated message. This is an active subcommand.

S Sends data bytes from the command string to the selected device. No response is required or accepted.

R Accepts response from selected device. The response will appear as a controller-generated message. This is an active command.

A Performs the IEEE-488 Interface Clear, and sets remote enable true.

In normal operation, only two commands seem to be important. The S command places the device in the desired mode of operation. The T command triggers it and takes a reading.

There are currently two devices on the bus: a Hewlett-Packard 3478A Multimeter, and an A. D. Data Systems 56D minimatrix. The address of the multimeter is 23. It is easy to use and the documentation is clear. The blue pages of its operator's manual give a quick guide to control from the instrumentation bus.

The switching matrix, at bus address 7, is also easy to use, but the documentation is less clear. The command strings take the form of three-digit numbers, concatenated with no spaces. The first digit is a command:

0 Turns the selected relay off.

1 Turns the selected relay on.

2 Turns all the relays off.

The second and third digits identify the selected relay. For example, the RWED command to clear all the relays and then turn on relays 1 and 13 would be:

IS7200101113

There are schemes that use these two instruments for evaluating the interconnect of a fresh RVLSI wafer, and for incrementally restructuring the interconnect, checking for certain kinds of faults.

The second laser table has become operational but lacks an autofocusing capability. RWED has been modified appropriately and is running on both tables. Currently, RWED is being used on the second system to position the laser over transistor sites for optical probing experiments. Among these are the debugging of defective LBS-generated MOSIS CMOS circuits.

E. PLACEMENT, ASSIGNMENT, LINKING

A set of RVLSI wafer design programs which were written for the 16-point FFT wafer in a cooperating program has been adapted for the packet radio integrator application. The FFT system, like the integrator, comprises a large number of cells of one type with a small number of peripheral cells on the wafer edges.

Four sets of files are created, either manually or automatically, to describe: (1) the logical wafer system, (2) the generic physical wafer, (3) a particular physical wafer, and (4) the actions necessary to restructure a wafer. Most of the files are ASCII files.

The signal nets of an RVLSI wafer are described in an "lnet" file. Programs exist to expand succinct descriptions of repetitive designs. At present, there is no link between these files and any other description of the system, such as a simulation. A "sys" file describes the topological arrays of cells comprising the system and also includes information about desired grouping of physical cells.

The generic physical wafer is described as a tiling of the wafer with cells. The integrator description requires 31 cell files, two for the counter cells and 29 for peripheral cells and interconnect at the edge of the array. All physical features are placed relative to an interconnect grid using an alphanumeric "picture" representation. During the early design stages the grid can be a virtual grid, and assignment and linking experiments may be done to assist in the design. Links at this stage may be logical connections only with the physical dimensions added later with a links file. A "wfr" file specifies the physical arrays of cells. The physical description is completed with the addition of dimensions to the cell and wfr files. A second physical description is required for the RWED system which is used to control the laser restructuring equipment. Display programs generate hard-copy and color CRT plots of the wafer with the virtual grid. Test results on the cells and interconnect are moved from the Tektronix S3260 tester to the VAX and massaged to generate a "prob" file description of a real wafer.

An assignment program assigns logical to good physical cells in order to produce a uniform distribution of spare cells within the grouping and ordering constraints in the "sys" file. Physical net list files, "pnet," are then generated and the "lsh" (link shell) program is used to generate interconnect paths. Paths are generated by searching along available tracks starting from all pins of a net. Cost functions, written into the program, favor the use of short track segments. If this search fails a more exhaustive search can be invoked. The user can influence the linking by ordering of the nets and by causing certain links to be invisible. The "lsh" program is operated interactively; manual intervention is supported, if necessary, and the graphics interactive facilities are very good.

The final stage in the restructuring design process is to generate command files for the laser restructuring equipment. A number of ordering and sorting options can be elected.

The packet radio integrator wafer has been described using these programs. The ability to work with a virtual grid in the earlier design stages was very valuable. The effort required to move the physical description from the physical design system (a CALMA interactive design system) was fairly large, as is the whole physical design effort. The description conventions work very well for signals but not as well for power connections due to the increased irregularity. The assignment and linking programs appear to be very effective for the integrator. This set of programs, along with the RWED programs, comprise a workable restructuring design system.

III. APPLICATIONS

A. DYNAMIC TIME WARPING (DTW) WAFER

The dynamic time warping (DTW) wafer is intended to solve the two dominant computational tasks in speech recognition: input/reference word template matching and input/reference word time registration. The DTW computation can be structured as a two-dimensional array of identical processing elements (PEs) where each PE receives data from the PEs immediately to its north, east, and northeast. The computation for a given input/reference comparison proceeds through the array in a "wavefront" traveling northeast. In other words a given diagonal's computation in the array depends on the calculations of the diagonal to the southwest. The goal of the DTW wafer is to exploit the parallelism and local interconnect properties in the computation outlined above in a restructurable wafer-scale implementation. Although it currently is not feasible to implement an entire DTW array (consisting typically of 500 to 1000 PEs) on a single wafer, a modular architecture has been developed where a subset of the array's diagonals are instantiated to process the entire array in a time serial fashion. A preliminary estimate is that 63 DTW PEs could be implemented on a wafer using Lincoln's 5-μm bulk CMOS process and would comprise approximately 350,000 functioning transistors. The wafer would typically be organized as two diagonals, 32 and 31 PEs wide, respectively, and would have a real-time throughput rate of 800 to 1600 input/reference word comparisons per second. Such a wafer could be used for a >100-word speaker independent isolated or connected word recognition system and represents an order of magnitude increase in computational power over commercially available systems.

An example of a 63-processing-element wafer organized as two diagonals (31 and 32 PEs, respectively) is shown in Fig. 1. The PEs will be implemented using bit-serial arithmetic. An artifact of this design approach is that a second type of element is required on the wafer consisting of pure delay. The delay elements are projected to comprise 15 percent of the functioning wafer circuits. Interconnect between PEs is local and

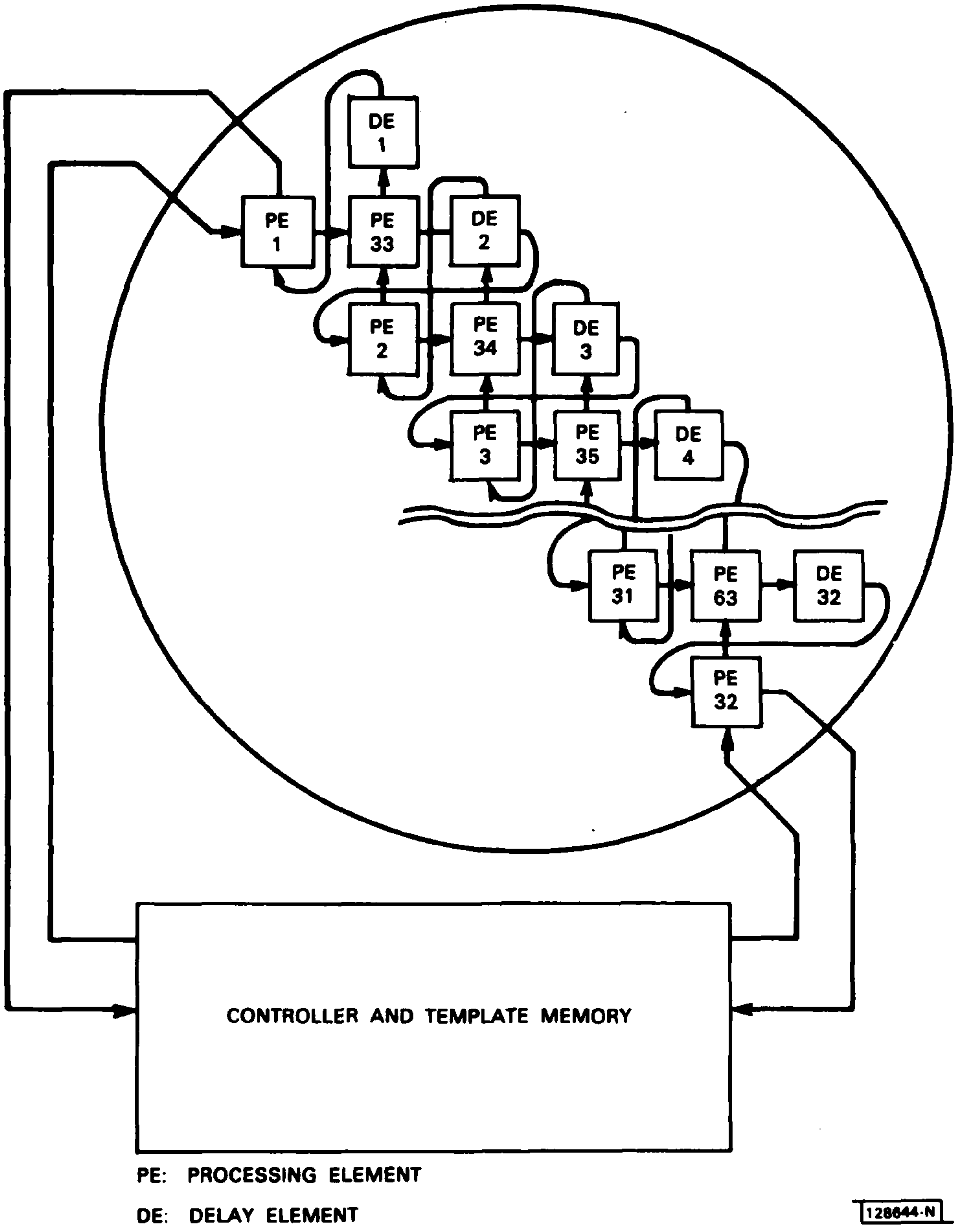


Fig. 1. DTW wafer with 63 processing elements.

relatively simple. Data communication outside the wafer is also elemental, consisting of connections only at the ends of the diagonal. A more general-purpose processor to control the wafer is envisioned to manage the higher level tasks in isolated and connected word recognition, such as template memory accessing and word distance comparisons.

The architecture for the processing element is shown in Fig. 2. It will be a special-purpose design based on serial arithmetic. Although this approach is less flexible than a programmable parallel arithmetic processing element, wafer yield considerations demand as small as possible an elemental unit and serial arithmetic is ideally suited to this. Furthermore, such an approach is expected to result in considerably higher throughput than more general-purpose techniques.

The PE itself consists of an input/reference frame distance computer and a time registration path computer. The distance computer derives a dissimilarity measure between input and reference template frames from the west, south, and southwest and passes them on to processors to the north, east, and northwest. The path computer uses this distance as well as path computations from previous diagonals to recursively update the best time registr3tion path to that PE. Two classes of distance measures are currently under consideration: LPC based and spectrally based. The computations for the LPC metric and the optimal path are:

```
Distance calculation:
    $D = 0$
    for $(0<i<10)$
        $\{D = D + I(i) * R(i)\}$

Path calculation:
    $S_{NE} = \text{minimum } (S_W+D, S_S+D, S_{SW}+2*D)$

  $S_N, S_E = S_{SW} + 2*D$
```

where R and I are the reference and input frame vectors.

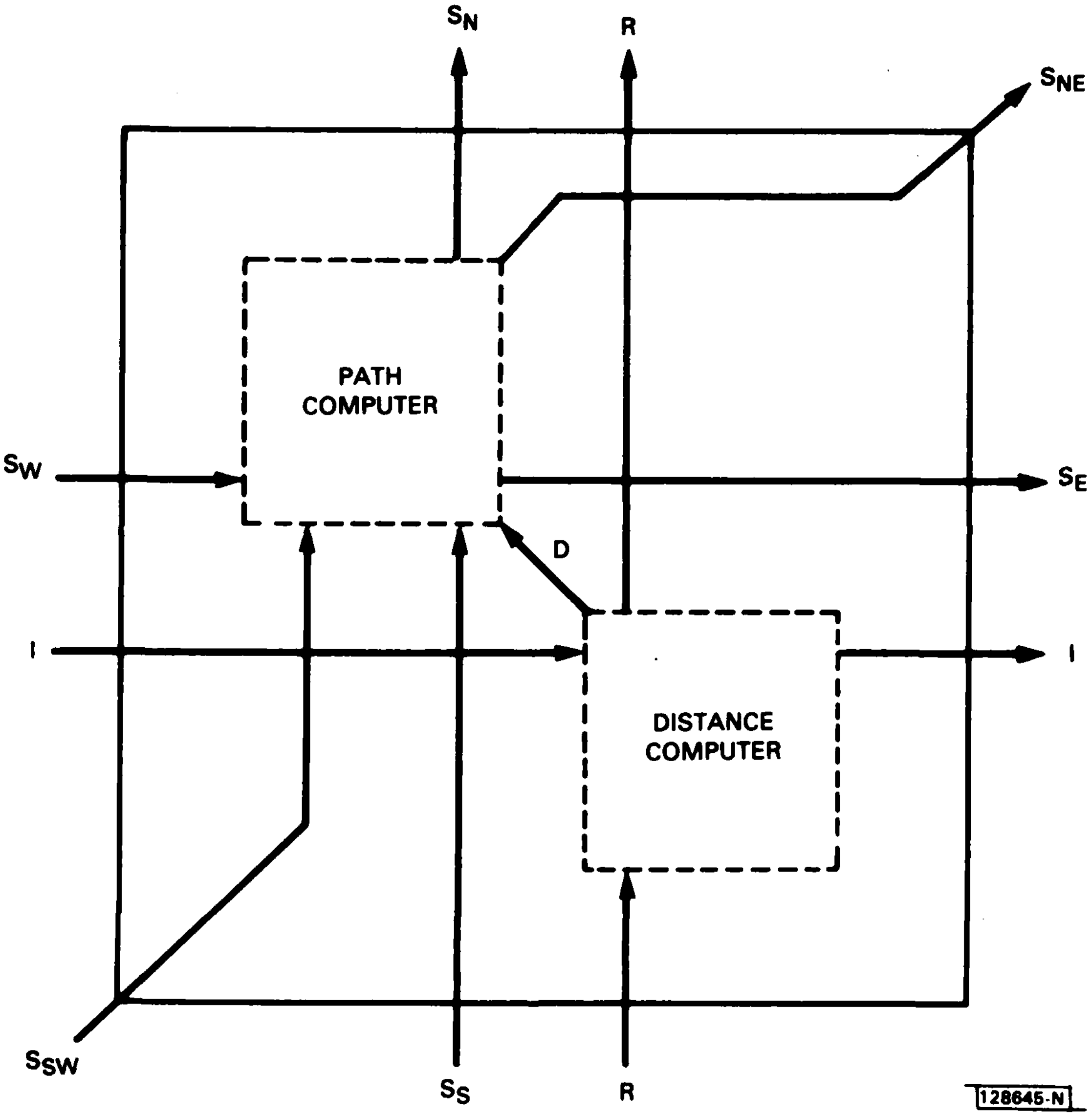


Fig. 2. DTW processing element.

A systolic system simulator has been written in LISP and is currently being used to develop a gate level simulation of the DTW wafer for detailed logical definition and verification of the architecture. A higher level simulation of the isolated and connected word recognition (level building) algorithm is also being developed in LISP as a further check on the architecture being defined with the systolic simulator.

Using bit-serial arithmetic techniques similar to those used in the 16-point FFT wafer being developed in a cooperative program, it is estimated that a total of 350,000 functioning transistors will be required for a 63-processor-element DTW wafer. These would be distributed as follows.

Circuit	Percentage of Transistors (percent)
63 processing elements	
63 distance computers	42
63 path computers	42
32 delay elements	16

B. AGC CIRCUIT

Due to the limited dynamic range and precision of fixed-point digital computers and analog-digtital (A/D) converters, narrowband speech compression devices (i.e., vocoders) operate most effectively over only a limited range of input speech volume levels. One solution to this problem is to increase the word length or use a floating-point format for the digital computation and A/D converter. In vocoder hardware implementations where size, power, and cost are considerations, this may not be possible. A more practical approach consists of an audio attenuator which precedes the vocoder A/D converter and conditions the input audio signal such that the dynamic range of the A/D converter and fixed-point computations is not exceeded. Such an approach has historically been implemented in analog automatic gain control (AGC) circuits. The following section describes a digitally controlled AGC

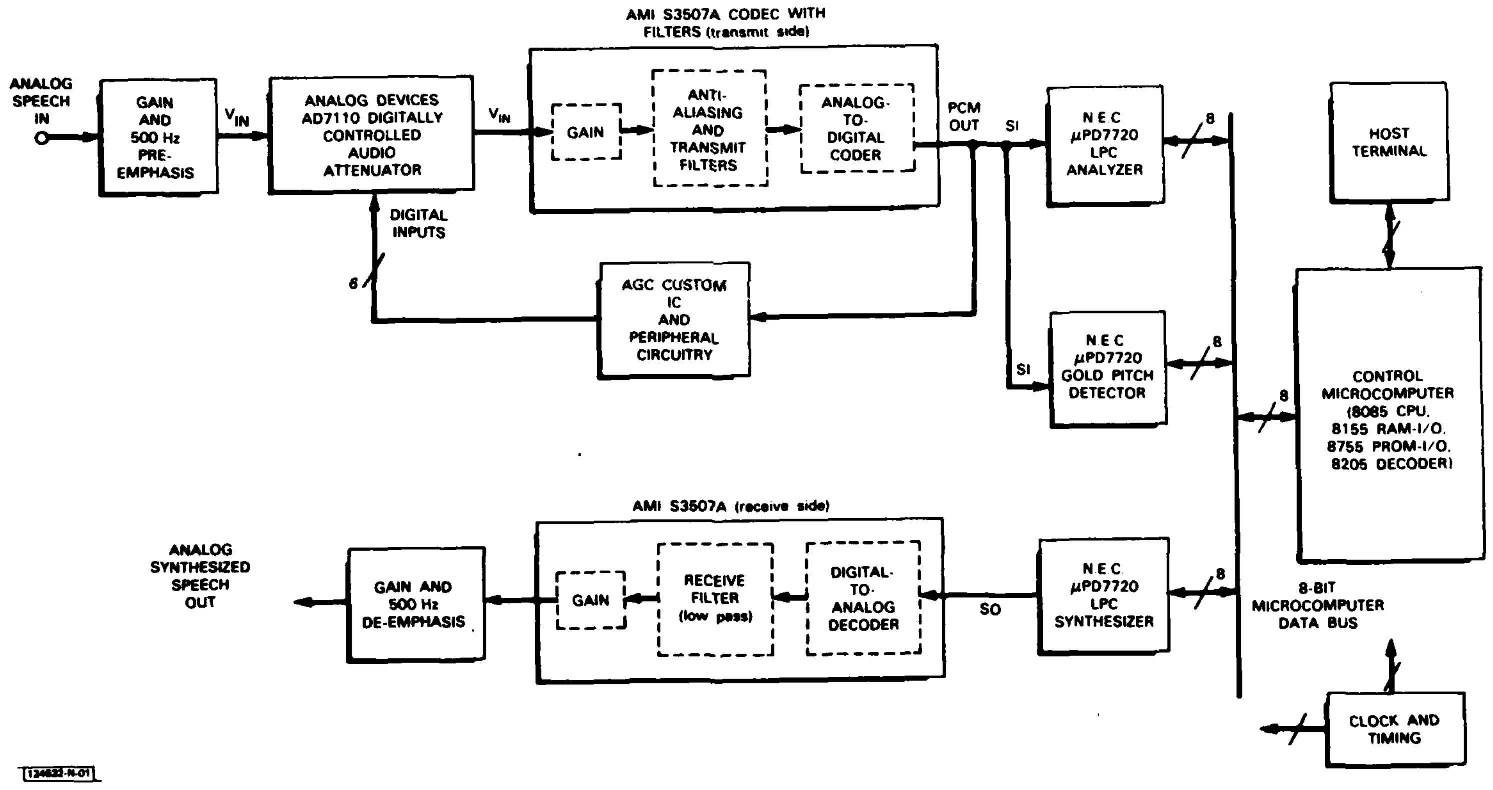

AMI S3507A CODEC WITH FILTERS (transmit side)
ANALOG SPEECH IN
GAIN AND 500 Hz PRE-EMPHASIS
V_{IN}
ANALOG DEVICES AD7110 DIGITALLY CONTROLLED AUDIO ATTENUATOR
V_{IN}
GAIN
ANTI-ALIASING AND TRANSMIT FILTERS
ANALOG-TO-DIGITAL CODER
PCM OUT
SI
N E C µPD7720 LPC ANALYZER
8
HOST TERMINAL
DIGITAL INPUTS
6
AGC CUSTOM IC AND PERIPHERAL CIRCUITRY
SI
N E C µPD7720 GOLD PITCH DETECTOR
8
8
CONTROL MICROCOMPUTER (8085 CPU, 8155 RAM-I/O, 8755 PROM-I/O, 8205 DECODER)
AMI S3507A (receive side)
ANALOG SYNTHESIZED SPEECH OUT
GAIN AND 500 Hz DE-EMPHASIS
GAIN
RECEIVE FILTER (low pass)
DIGITAL-TO-ANALOG DECODER
SO
N E C µPD7720 LPC SYNTHESIZER
8
8-BIT MICROCOMPUTER DATA BUS
CLOCK AND TIMING
134632-N-01

Fig. 3. LPC vocoder with AGC.

technique which is considerably more flexible than traditional analog techniques.

The digitally controlled AGC is based on the Analog Devices AD7110 digitally controlled audio attenuator which is placed after the vocoder audio input pre-emphasis filter, and before the anti-aliasing filter and A/D converter assembly (Fig. 3). The digital controller for the attenuator monitors the digital output from the A/D converter on a sample-by-sample basis and, in concert with vocoder frame timing, outputs a digital attenuation value.

The controller is based on a "fast-attack/slow-decay" type of algorithm. The attenuation increases rapidly when the A/D converter is observed to be saturating. When the input level is too low the attenuation value is decreased gradually. Constraints are placed on the speed at which the attenuation is decreased in order not to respond to normally occurring intonations and pauses in speech; otherwise, unacceptable distortion would result. Finally, to maintain the integrity of the vocoder analysis, changes in the digital attentuation value occur only at frame boundaries.

A custom NMOS integrated circuit to implement the digital control of the AGC technique has been designed using the MACPITTS silicon compiler.[1] The MACPITTS compiler accepts a high-level algorithmic description as input to produce the mask descriptions required for MOSIS integrated circuit fabrication. The quick design cycle achievable with the MACPITTS program is demonstrated by the fact that the AGC design was completed in approximately 3 weeks by a digital systems designer previously unfamiliar with the MACPITTS design tool or IC design.

The AGC algorithm tasks are partitioned into those performed once each sampling interval (typically 125 μs) and those performed once per speech frame (typically 22.5 ms). The sample-rate tasks begin with the input of a μ-255 law PCM speech sample (μ-law A/D coder) and updating a register which holds the maximum PCM value seen during the current frame. Another register maintains a count of the number of input samples which have saturated the A/D coder during the current frame.

The frame-rate processing compares the maximum PCM value seen over the past frame in the sample-rate task to several thresholds and decides whether to increase, decrease or maintain the current attenuation value. If the maximum input value saturated the A/D coder, then the "fast/attack" case is involved and the attenuation is increased by 1.5 dB (one binary code step). If several samples saturated the A/D coder during the past frame, then more drastic measures are taken and the attenuation is increased by a total of 3 dB. If the maximum value seen during the previous frame is more than 6 dB below saturation, a potential "slow-decay" case is signaled. In order to avoid responding to normally occurring pauses or intonations in speech, the frame-rate processing keeps a count of consecutive frames in which the maximum value to the A/D coder was less than 6 dB below saturation. The count is incremented once for each frame (typically 25 frames or about a one-half second of speech), the attenuation is decreased by 1.5 dB and the decay counter is reset. Since consecutive low-level frames are required, any frame whose maximum value was larger than 6 dB below saturation will also reset this decay counter. When the maximum value seen in the past frame is more than 30 dB below the A/D coder saturation value, the frame is declared silent and no attentuation value adaptation is allowed. In addition, the decay count is not affected.

The AGC IC also has a "push-to-talk" input line for use in half-duplex vocoder systems. When asserted, this line has the same effect as silence frames in that no adaptation of the attentuation value is permitted. In all frames, the frame-rate processing concludes by resetting the maximum value and saturation count registers used by the sample-rate processing.

The AGC algorithm described above has been implemented in a custom 4-μm linewidth NMOS IC designed with the MACPITTS silicon compiler. Since the MACPITTS user requires minimal knowledge of integrated circuit design and since the MACPITTS design language is highly readable (e.g., a structured language similar to such high-level programming languages as LISP), IC designs like the AGC circuit can be completed in a relatively short time by a potentially large community of users.

The MACPITTS compiler uses a target architecture based on a combinatorial control section implemented as a Weinberger array[5] and a data path consisting of registers, combinatorial functional units, and multiplexers for interconnect between the data path units. A floor plan of the AGC chip is shown in Fig. 4 depicting the control and data path regions. In the case of the AGC IC an 8-bit data path is specified including four registers, two subtracters, one incrementer, a 1's complement operator, and four 1-bit flags. The remaining space in the data path consists of the multiplexers which output to the registers from the functional units. The chip uses 2190 N-channel MOS transistors occupying 4.5 × 5.4 mm^2.

The AGC chip was fabricated by the DARPA MOSIS silicon foundry. Of 9 chips fabricated, all passed functional tests. These tests were run at 400 kHz, which is double the required system clocking rate. More detailed performance tests are beginning. A photomicrograph of the AGC chip can be seen in Fig. 5.

The pin-out of the AGC IC as well as the peripheral circuitry necessary for use in the LPC vocoder audio input path are shown in Fig. 6. The μ-255 law PCM data are input on 8 input lines and the digital attenuation value is output on 6 lines. The remaining input lines include the power, ground, clock, reset, push-to-talk, and frame and sample time strobes. In addition, twelve internal values are brought out for prototype test purposes. The peripheral circuitry includes an 8-bit serial-to-parallel converter and latch for the PCM data from the CODEC-with-filters chip, a TTL-to-CMOS level converter for input of the attenuation value to the audio attenuator, and a 3-phase clock generator.

The timing of data input/output and sample-rate ("sample-stuff") and frame-rate ("frame-stuff") processing is shown in Fig. 7. The sample-strobe rising edge initiates the sample-rate processing which, as described above, begins with input of the PCM sample. Similarly, the frame-strobe initiates the frame-rate processing which concludes with the output of a new attenuation value if necessary. Since throughput was not a driving

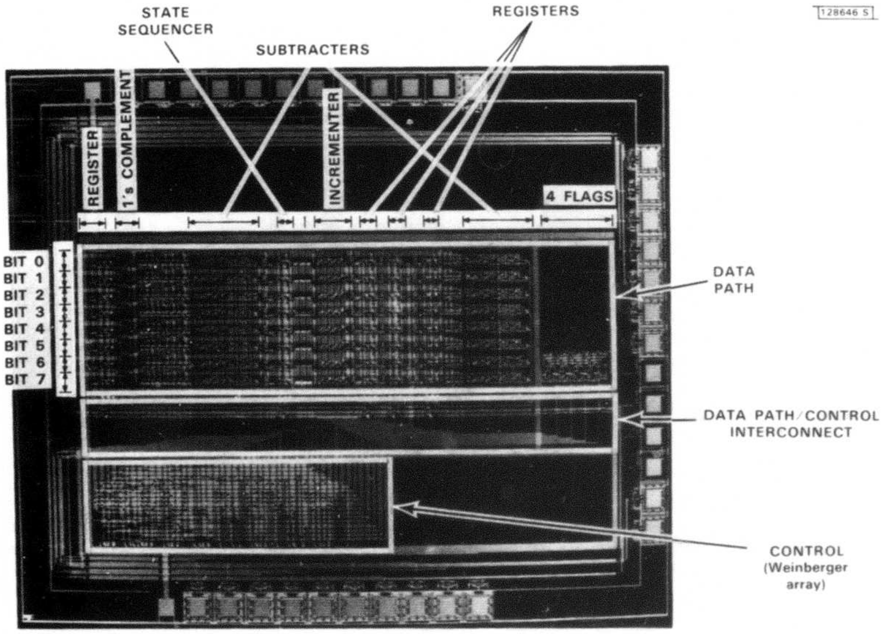


Fig. 4. AGC IC floor plan.

128647 S

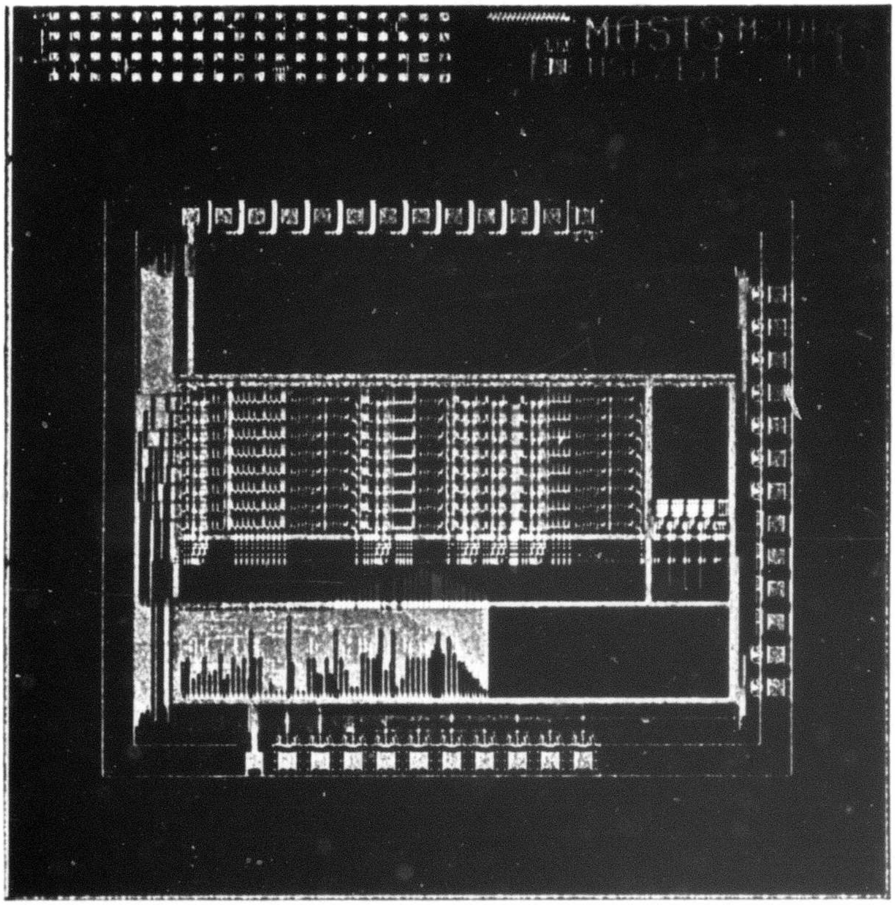


Fig. 5. Photomicrograph of AGC chip.

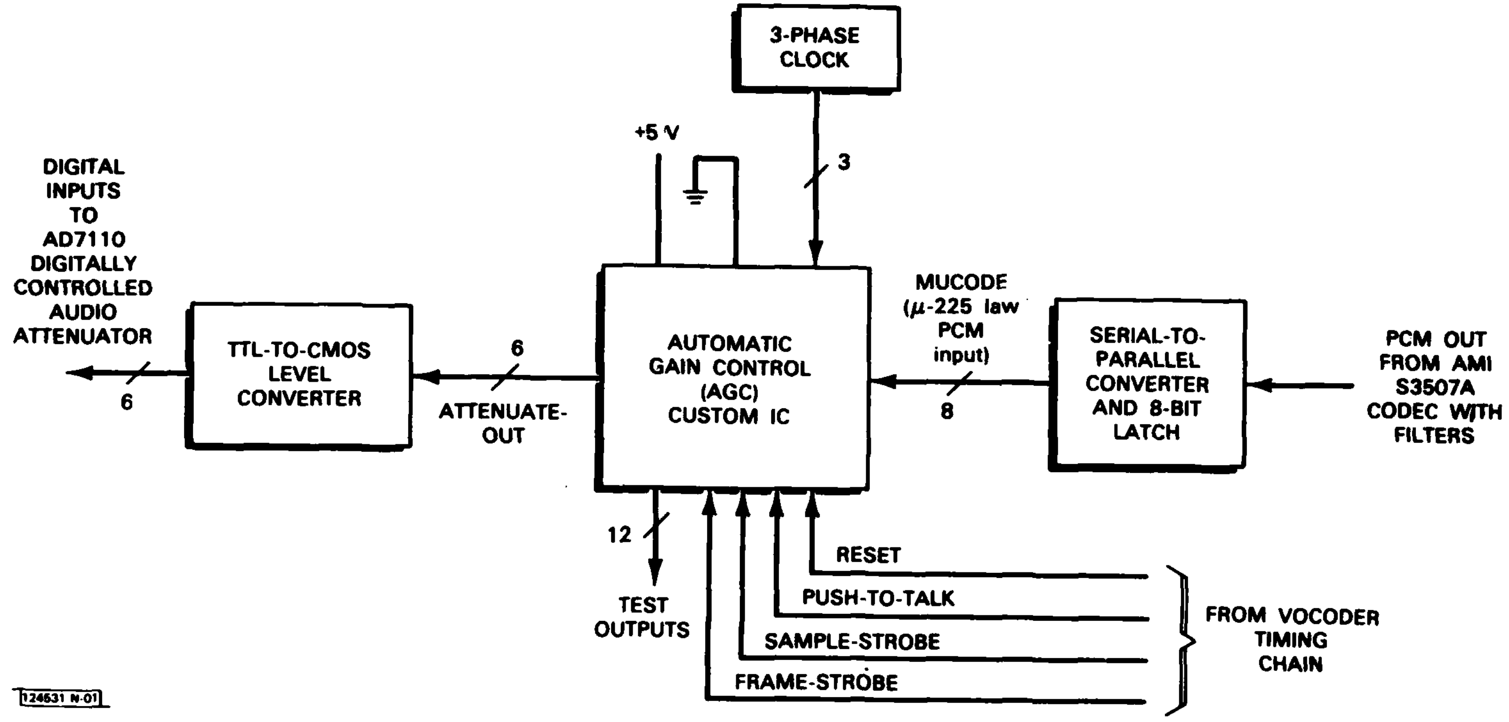

3-PHASE CLOCK
3
+5 V
DIGITAL INPUTS TO AD7110 DIGITALLY CONTROLLED AUDIO ATTENUATOR
6
TTL-TO-CMOS LEVEL CONVERTER
6
ATTENUATE-OUT
AUTOMATIC GAIN CONTROL (AGC) CUSTOM IC
MUCODE (μ-225 law PCM input)
8
SERIAL-TO-PARALLEL CONVERTER AND 8-BIT LATCH
PCM OUT FROM AMI S3507A CODEC WITH FILTERS
12
TEST OUTPUTS
RESET
PUSH-TO-TALK
SAMPLE-STROBE
FRAME-STROBE
FROM VOCODER TIMING CHAIN
124531 N-01

Fig. 6. AGC IC pin-out and peripheral circuitry for use in the LPC vocoder.

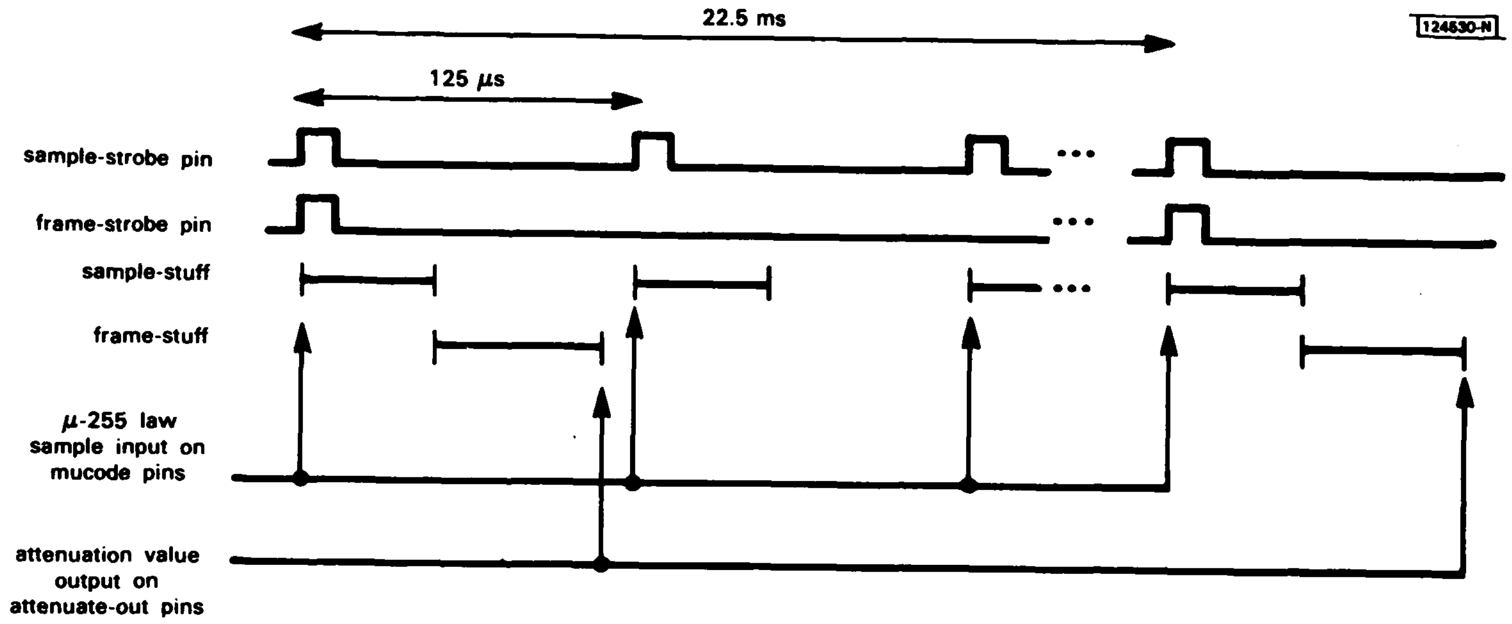
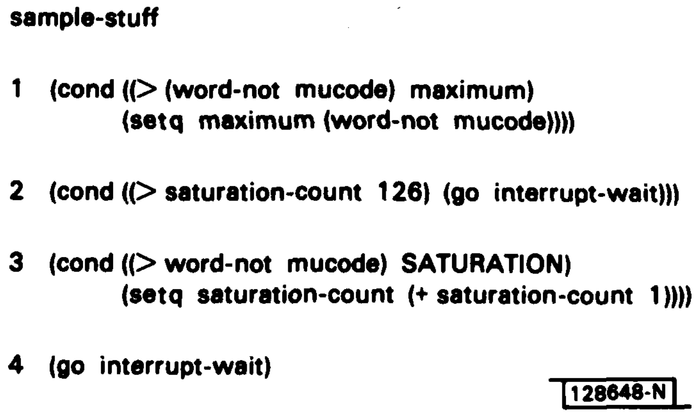

22.5 ms
124630-N
125 μs
sample-strobe pin
frame-strobe pin
sample-stuff
frame-stuff
μ-255 law sample input on mucode pins
attenuation value output on attenuate-out pins

Fig. 7. AGC IC input/output sample-rate ("sample-stuff") and frame-rate ("frame-stuff") processing timing.

constraint in this design, the AGC IC was designed to do the sample-rate and frame-rate processing in a time-serial fashion. Arbitration between sample- and frame-strobes is achieved by storing the rising edge occurrences in separate flags and using a polling mechanism to service them. A benefit of this technique is that sample- and frame-strobes only need to be loosely synchronized in time.

A slightly simplified section of the AGC source code is provided (Fig. 8) to give an example of the nature of circuit design using the MACPITTS compiler. The code segment shown here corresponds to the sample-rate processing described above. A flowchart documenting this section is given in Fig. 9. Each line in the fragment is executed sequentially, one per AGC IC main clock cycle, and is numbered for explanatory purposes. In line 1 the 1's complement of "mucode" (the µ-255 PCM input) is compared to the value of the internal register "maximum" which holds the largest magnitude input seen in the current frame. If mucode exceeds maximum, then maximum is updated. Lines 2 to 4 update the register "saturation-count" which counts the number of samples in the current frame which have saturated the A/D



```
sample-stuff

1  (cond ((> (word-not mucode) maximum)
         (setq maximum (word-not mucode))))

2  (cond ((> saturation-count 126) (go interrupt-wait)))

3  (cond ((> word-not mucode) SATURATION)
         (setq saturation-count (+ saturation-count 1))))

4  (go interrupt-wait)
```

Fig. 8. Fragment of AGC MACPITTS source code.

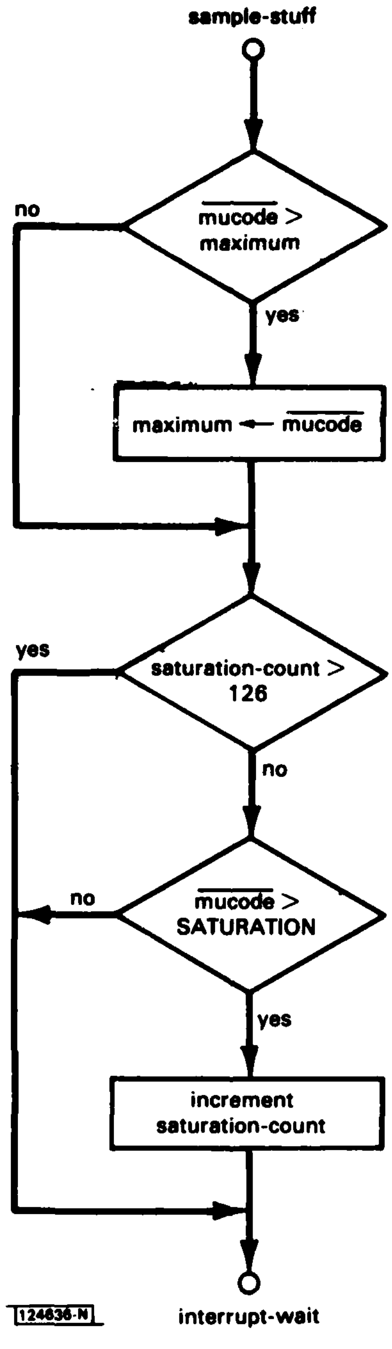

sample-stuff
$\overline{\text{mucode}}$ > maximum
no
yes
maximum ← $\overline{\text{mucode}}$
saturation-count > 126
yes
no
$\overline{\text{mucode}}$ > SATURATION
no
yes
increment saturation-count
124636-N
interrupt-wait

Fig. 9. Flowchart for sample-rate task of AGC algorithm.

coder. If saturation-count is at its maximum value (127) then control returns to a loop ("interrupt-wait") which polls for sample- and frame-strobes (line 2). Next, in line 3, if mucode is greater than the A/D code saturation threshold, "SATURATION," then saturation-count is incremented. This concludes the sample-rate task and control returns to the interrupt-wait loop (line 4). The registers "mucode" and "saturation-count" were explicitly declared earlier in the program and account for two of the four registers indicated in the data path of Fig. 4. The implicit comparisons and incrementers result in the explicit instantiation by the MACPITTS compiler of the subtracters and the incrementers in the data path. The entire MACPITTS AGC chip specification contains approximately 90 lines of code as seen in Appendix B.

A very compact LPC vocoder[6] has been adapted to include the AGC technique outlined above. In this system (Fig. 3) the analog input speech is processed through the Analog Devices AD7110 digitally controlled audio attenuator. The AD7110 provides 88.5 dB of attenuation in 1.5-dB steps determined by a 6-bit binary input code. The analog speech is then converted to an 8-bit μ-255 law PCM code by the transmit half of the AMI S3507A CODEC-with-filters chip and transmitted serially to the digital analysis portion of the vocoder. The PCM code is also fed back to the custom AGC integrated circuit. The AGC IC uses these data as well as vocoder frame timing information to update the audio attenuator digital inputs. The LPC vocoder digital analysis (linear predictive analysis and pitch/voicing estimation) is implemented with two N.E.C. μPD7720 Signal Processing Interface (SPI) single-chip microcomputers.[6] The LPC synthesis is realized with a third SPI device while an Intel 8085 minimum configuration 8-bit microprocessor is used for control and communication between the three SPIs and the outside world (host terminal).

IV. TESTING

The TOC system, described in the last Semiannual Technical Summary,[7] is a low-cost functional IC tester, consisting of an array of four-pin slices, together with a small amount of common interface and control circuitry. For testing dynamic circuitry, there is a provision for looping through a hold sequence using one memory bank while the other is being reloaded.

Three possible versions of the TOC architecture were investigated, all potentially implementable using MACPITTS. The original design applied a vector to the device under test every ten MACPITTS clock cycles. Three cycles were used for the vector fetch. During each of the next four cycles, a single pin of each slice was tested against the expected value. The last three cycles of the loop were used to write the resultant vector back out to memory. In this way two partitions of a single physical memory were used for both logical banks.

The second design considered was more ambitious; vector sequencing was to occur at the rate of once every cycle. To accomplish this, two physical memories would be used. Each cycle, a vector would be latched both from the source memory and the device under test. During that same cycle, the result of the last vector would be written to the destination memory. Unfortunately, the pin count and system size for this design were judged unwieldy.

The design approach ultimately chosen is a compromise. During the even cycles, a vector is latched from memory and the device-under-test signals are latched as well. Then, during the odd cycles, an error in the previous vector is written back out to memory. If there is no error on a given cycle, that cycle may be stolen for reading vectors from the last test or writing vectors for the next test. This design affords the same economies of memory and pin count as the original design, yet with five times greater throughput.

Figure 10 is a block diagram of the current TOC system design. The system consists of a Western Digital TR1865 UART, a small amount of clock generation circuitry, and as many four-pin cascadable tester slices as are

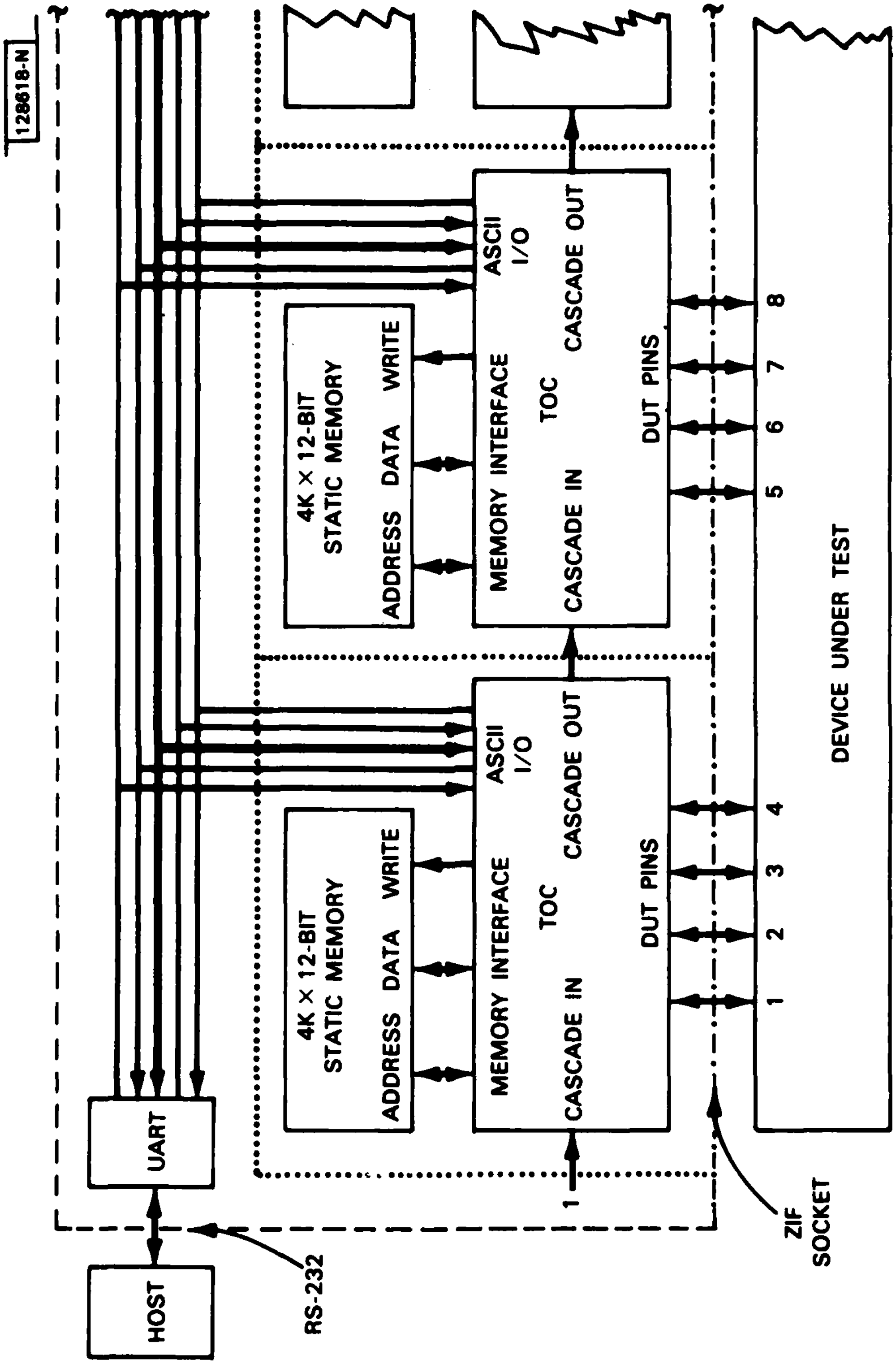


Fig. 10. TOC system block diagram.

desired. Each slice is one custom TOC IC and three INMOS IMS1420-55 4K × 4-bit static memory circuits.

Figure 11 shows how the TOC chip organizes this 4K x 12-bit memory into two banks of 2K × 12 bits. The banks are used to hold both the vectors to, and the responses from, the device under test. One bank is loaded by the host while the other drives the test sequence. The arrows in Fig. 11 show the path through the memory taken by the vector sequencer:

(a) Characters from the host, containing test vectors in octal format, are loaded into first memory bank of each slice. The vector list contains a one-time test, followed by a repeated portion. This second part can be used to keep dynamic circuits "alive." An error during the one-time test, or an error in any pass of the dynamic holding loop will be reported.

(b) The "go" command is issued by the host, and the testing sequence proceeds. Vectors in which errors occurred are rewritten with the comparison bit reset. TOC will transmit the success or failure of the previous test back to the host.

(c) During this test and its subsequent hold loop, the host is free to examine the alternate memory bank to find out where errors were, and then load it with the next set of vectors.

(d) When the "go" command is reissued, the memory banks switch roles, and the sequence continues with step (b).

Figure 12 gives the encoding TOC uses for vectors in both its internal data path, and the memory banks. Each octal digit contains the information associated with a single pin of the device under test. A one in the force bit causes TOC to drive the DUT pin with the value given in the data bit. A one in the compare bit causes TOC to test the DUT pin against the value specified in the data bit. This bit is reset to indicate an error.

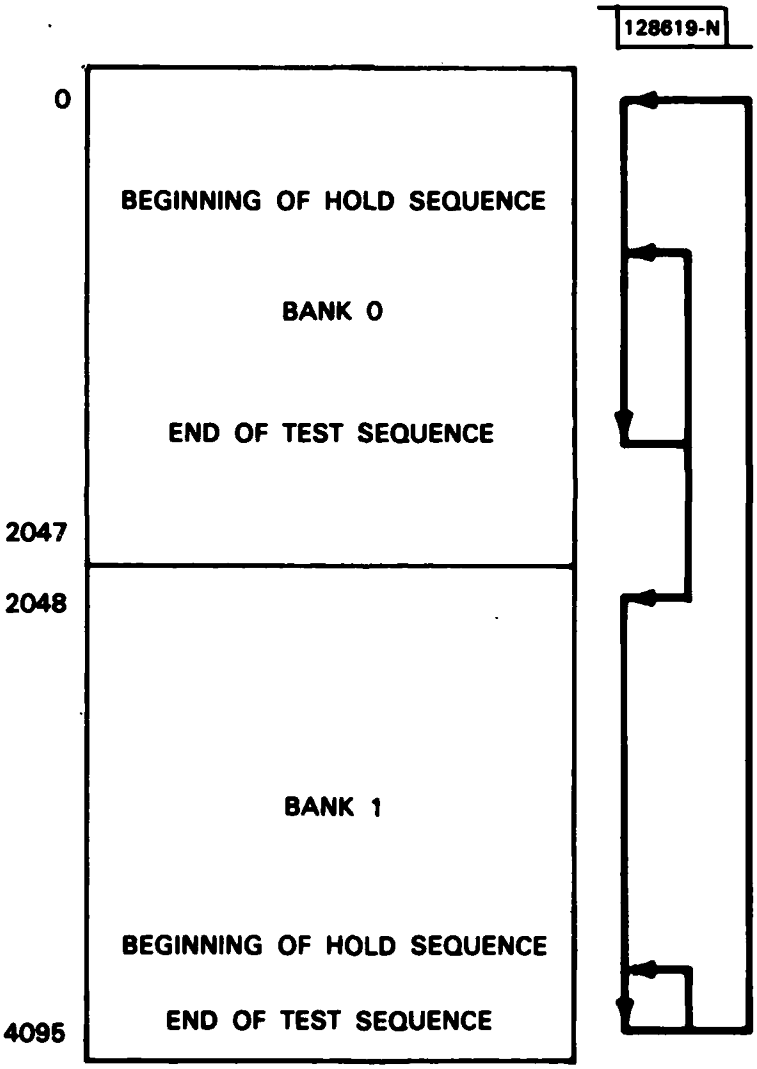


Fig. 11. TOC memory banks.

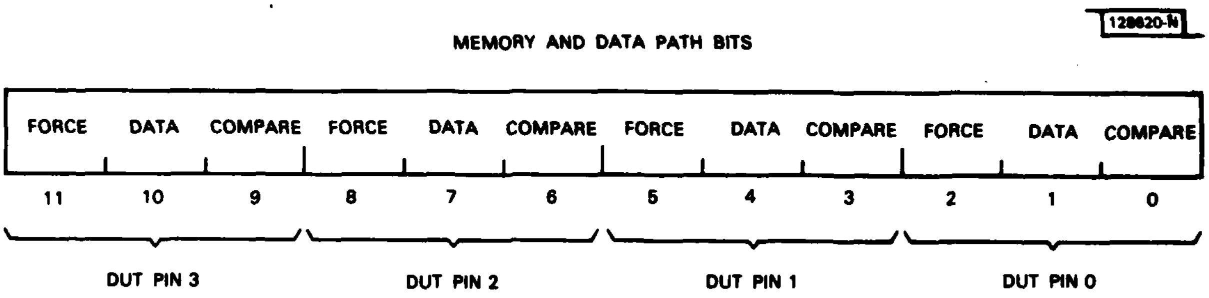


Fig. 12. TOC vector encoding.

Figure 13 illustrates how the TOC chip fetches vectors from its memory bank and updates according to the response. During the even cycles, the next vector is fetched and applied to the device under test. Simultaneously, the previous vector is shifted into Vector 2, and the response associated with it is latched into Vector 3. Then, during the odd cycles, one of two things happens: either there is an error, and the offending vector is written back out to memory with the comparison bit cleared, or the cycle is free for a transfer between the host and the other memory bank.

The MACPITTS description of TOC comprises about 200 lines of code. When originally compiled, TOC was too large to fabricate in MOSIS (9.0 × 10.5 mm). Much of the recent work on MACPITTS, particularly the second pass (layout portion), has been directed toward the goal of reducing the size of TOC. The efforts of major importance included:

(a) A 3-μm pad library was created and installed. This allowed moving to 3-μm technology.

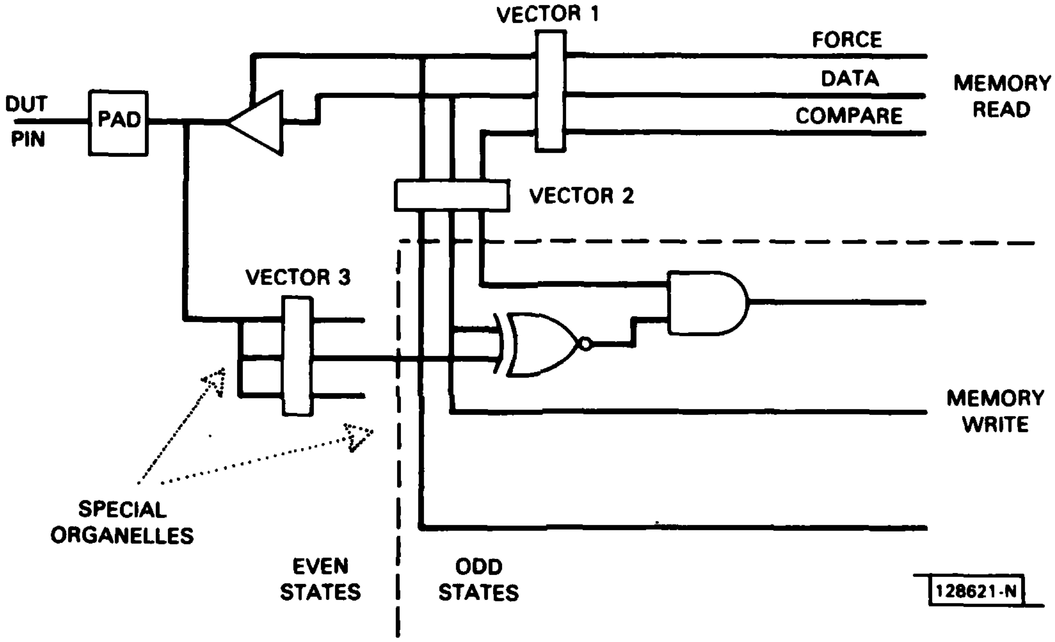


Fig. 13. TOC data flow.

(b) The register unit was laid out again by hand using buried contacts. There are 13 registers in TOC.

(c) Several of the other organelles were laid out more efficiently, again using buried contacts.

Although the TOC MACPITTS description remained substantially unmodified throughout, it was recently altered to include two new organelles. These TOC-specific circuits were designed and laid out by hand in about two weeks. They replaced two XOR units, two AND units, one OR unit, one shifter, six internal ports, and one bit unit for a total of 13 of the 42 data path units. These two custom organelles and their role in the TOC data path are shown in Fig. 13.

TOC was successfully switch-level simulated using "nl" early on. This "benchmark" simulation was then used to verify the changes to MACPITTS and TOC as work progressed. The finished design was design rule checked using the Lincoln MDRC system[8] with a new buried contact rule script.

The effort resulted in a design 7.2 × 6.4 mm in 3 μm, small enough to fabricate. Figure 14 is a photograph of the design. It contains 4810 transistors and has a predicted power dissipation of 1 W. It is being fabricated in the 21 March MOSIS 3-μm test run.

Future plans include the prototyping of a complete TOC system and development of high-level software support for functional testing. A testing methodology has been proposed that is particularly suited to designs for which there is already a switch-level simulation. Instead of having the designer generate a new behavioral description for testing purposes, he can simply drive the tester with the same, or a possibly enhanced version of the simulation used to verify the design. For "nl," this can be implemented easily by rewriting the clock routine to output an appropriately formatted vector file. There are also plans to accommodate other forms of test vector generation, possibly ICTEST, and vector formats used on the Tektronix 3260.

128622 S

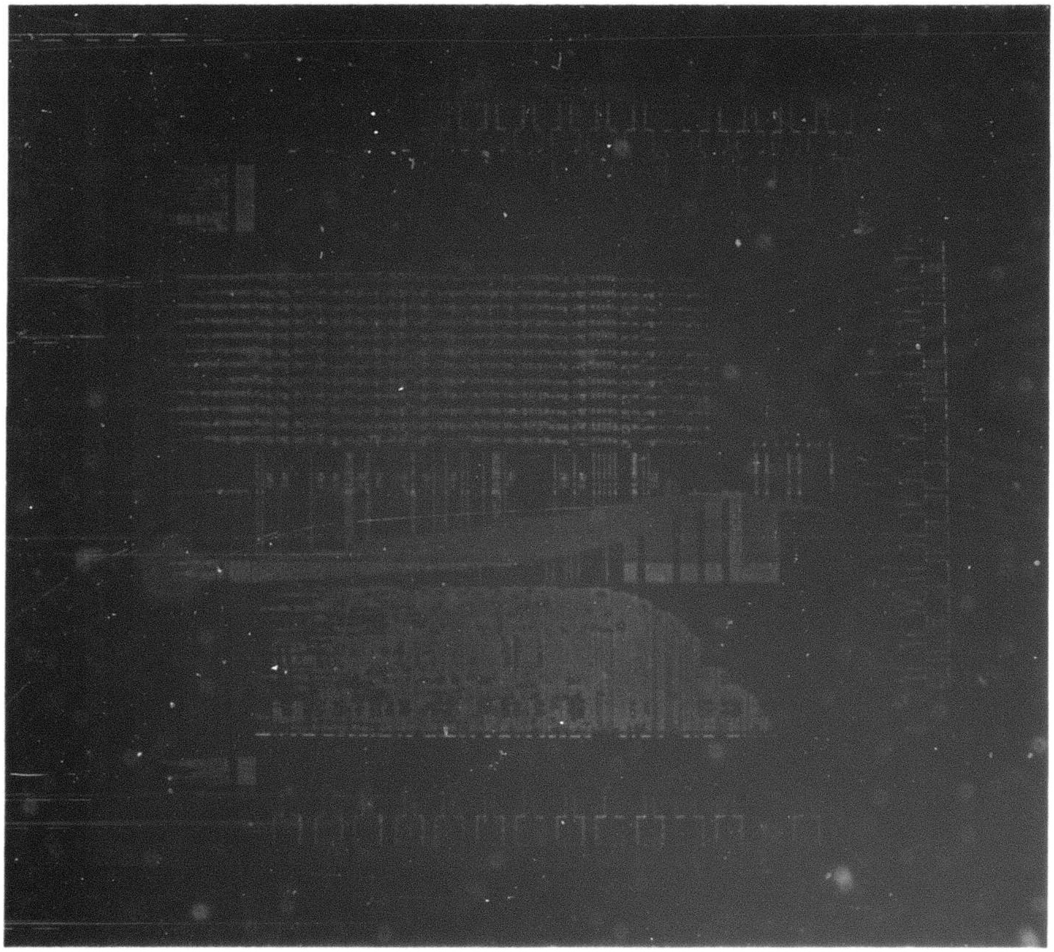


Fig. 14. TOC four-pin functional tester slice.

REFERENCES

1. J.M. Siskind, J.R. Southard, and K.W. Crouch, "Generating Custom High Performance VLSI Designs from Succinct Algorithmic Descriptions," Proc. Conf. on Advanced Research in VLSI, Cambridge, Massachusetts, 25-27 January 1982, pp. 28-40.

2. Restructurable VLSI Program Semiannual Technical Summary, Lincoln Laboratory, M.I.T. (30 September 1981), p. 13, DTIC AD-A112037/7.

3. J.K. Ousterhout, "CAESAR: An Interactive Editor for VLSI Layout," VLSI Design (Fourth Quarter, 1981).

4. C.A. Mead and L.A. Conway, Introduction to VLSI Systems (Addison-Wesley, Reading, Massachusetts, 1980).

5. A. Weinberger, "A Large Scale Integration of MOS Complex Logic: A Layout Method," IEEE JSSC SC-2, 182-190 (December 1967).

6. J.A. Feldman and E.M. Hofstetter, "A Compact, Flexible LPC Vocoder Based on a Commercial Signal Processing Microcomputer," Electro '82, Boston, 25-27 May 1982, Session 22/5.

7. Restructurable VLSI Semiannual Technical Summary, Lincoln Laboratory, M.I.T. (30 September 1982), DTIC AD-A126216.

8. A.J. Giovinazzo, "A Mask Design Rule Checking System," Proc. Intl. Conf. on Circuits and Computers (October 1980).

APPENDIX A

BNF FOR LBS TOPOLOGICAL INTERMEDIATE FORM

```
<topological intermediate specification>:={(<chan-type>(<sform>*))}
<chan-type>:=p|n
<sform>:=(<type><strap><op>(<tracki>*))
<type>:=(in name|out name|gate name)
<strap>:=(<tracki> from#to#)
<op>:=par|ser|pass
<tracki>:= column#
```

APPENDIX B

MACPITTS PROGRAM FOR AGC CIRCUIT

```
(program agc B
 (def 37 power)
 (def 1 ground)
 (def 2 phic)
 (def 3 phia)
 (def 4 phib)
 (def mucode port input (5 6 7 8 9 10 11 12))
 (def attenuate-out port output (13 14 15 16 17 18 19 20))
 (def reset signal input 21)
 (def sample-strobe signal input 22)
 (def frame-strobe signal input 23)
 (def push-to-talk signal input 24)
 (def max-out port output (25 26 27 28 29 30 31 32)) ; for testing
 (def sam-int-out signal output 33) ; for testing
 (def frm-int-out signal output 34) ; for testing
 (def sam-wait-out signal output 35) ; for testing
 (def frm-wait-out signal output 36) ; for testing
 (def sample-interrupt flag)
 (def frame-interrupt flag)
 (def sample-wait flag)
 (def frame-wait flag)
 (def attenuate register)
 (def maximum register)
 (def decay-count register)
 (def saturation-count register)
 (def SATURATION constant 126) ; +- .968, mu-255 code: x 000 0001
 (def SILENCEO constant 48) ; +- ~.03125, mu-255 code: x 100 1111 (- 30 db)
 (def SATNUMBER constant 12)
 (def DECAY-NUMBER constant 25) ; 25 frames or about 1.5 dB/.5 sec
                                 ;    decay rate
 (def OKAY constant 112) ; +- 0.5, mu-255 code: x 000 1111 (- 6 db)
 (def ATTENUATED constant 16) ; mid-way attenuation value

; sample interrupt edge detector
 (always
  (cond (sample-wait
         (cond (sample-strobe (setq sample-interrupt t)
                              (setq sample-wait f))))
        ((not sample-strobe) (setq sample-wait t))))

; frame interrupt edge detector
 (always
  (cond (frame-wait
         (cond (frame-strobe (setq frame-interrupt t) (setq frame-wait f)))
        ((not frame-strobe) (setq frame-wait t))))
 (always
       (setq max-out maximum) ; for testing
       (setq sam-int-out sample-interrupt) ; for testing
       (setq sam-wait-out sample-wait) ; for testing
       (setq frm-int-out frame-interrupt) ; for testing
       (setq frm-wait-out frame-wait) ; for testing
       (setq attenuate-out attenuate))
(process stuff 0
 initialize
 (par (setq sample-wait t)
      (setq sample-interrupt f)
      (setq frame-wait t)
      (setq frame-interrupt f)
      (setq attenuate ATTENUATEO)
      (setq decay-count 0)
      (go frame-reset))

 interrupt-wait
       (cond (sample-interrupt)
               (setq sample-interrupt f) (go sample-stuff)))
       (cond (frame-interrupt
               (setq frame-interrupt f) (go frame-stuff)))
       (go interrupt-wait)

 sample-stuff
       (cond ((unsigned-> (word-not mucode) maximum)
              (setq maximum (word-not mucode))))
       (cond ((unsigned-> saturation-count 126)
              (go interrupt-wait)))  ;overfl. possible
       (cond ((unsigned-> (word-not mucode) SATURATION)
              (setq saturation-count (- saturation-count -1))))
       (go interrupt-wait)

 frame-stuff
       (cond ((and (unsigned-> maximum SILENCEO) push-to-talk))
             (t (go frame-reset)))
       (cond ((unsigned-> OKAY maximum))
             (t (setq decay-count 0) (go check-saturation)))

 check-decay
       (cond ((unsigned-> decay-count DECAY-NUMBER)
              (setq decay-count 0))
             (t (setq decay-count (- decay-count -1)) (go frame-reset)))
       (cond ((unsigned-> attenuate 0)
              (setq attenuate (- attenuate 1))
              (go frame-reset))
             (t (go frame-reset)))
```

```
check-saturation
      (cond ((unsigned-> attenuate 30) (go frame-reset)))
      (cond ((unsigned-> saturation-count 0)
             (setq attenuate (- attenuate -1)))
            (t (go frame-reset)))
      (cond ((unsigned-> attenuate 30) (go frame-reset)))
      (cond ((unsigned-> saturation-count SATNUMBER)
             (setq attenuate (- attenuate -1))))

frame-reset
      (par
              (setq saturation-count 0)
              (setq maximum 0)
              (go interrupt-wait))))
```

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
ESD-TR-83-024

2. GOVT ACCESSION NO.
AD-A131246

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
Restructurable VLSI Program

5. TYPE OF REPORT & PERIOD COVERED
Semiannual Technical Summary
1 October 1982 — 31 March 1983

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Peter E. Blankenship

8. CONTRACT OR GRANT NUMBER(s)
F19628-80-C-0002

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Lincoln Laboratory, M.I.T.
P.O. Box 73
Lexington, MA 02173-0073

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
ARPA Order 3797
Program Element No. 61101E
Project No. 3D30

11. CONTROLLING OFFICE NAME AND ADDRESS
Defense Advanced Research Projects Agency
1400 Wilson Boulevard
Arlington, VA 22209

12. REPORT DATE
31 March 1983

13. NUMBER OF PAGES
62

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
Electronic Systems Division
Hanscom AFB, MA 01731

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

None

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

VLSI
Restructurable VLSI (RVLSI)
programmable interconnect
defect avoidance
customization
hardware description language
placement
routing
systolic array
integrator

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes work performed on the Restructurable VLSI Program sponsored by the Information Processing Techniques Office of the Defense Advanced Research Projects Agency during the semiannual period 1 October 1982 through 31 March 1983.